Rand
SANTA MONICA, CA 90406

File
SAFE

82-0538/12

April 22, 1982

Mr. Harry Fitzwater
Deputy Director/Administration
Central Intelligence Agency
Washington, D.C. 20505

Dear Harry:

Subsequent to our meeting on Tuesday, the 13th, Ron Wigington, [redacted] [redacted] and I spent about an hour together at NSA. Ron and I also talked about matters at dinner after the meeting. The following views represent the composite position of the three of us; a draft was circulated for review and comment prior to this final letter. All four, however, have been copied.

We believe it important for us to be present at the time TRW makes its presentation on the four tasks that you have levied on them. It will be important to judge how seriously the effort has been taken; it will be important to judge whether the investment everyone agrees has been made can be recaptured or is in fact of value; it will be important to estimate the quality of any briefers who may be new players. Our feeling is that first hand exposure is important for these and other reasons; too much will be lost through 2nd party filtering.

We think that an internal group should work concurrently to examine other and hopefully more contemporary innovative architectures. In this regard, certainly [redacted] is an enormously valuable asset; I'm told that a Dan Wallace could be as well. There are undoubtedly others that we haven't met. This suggestion stems from the observation that the commercial world is vigorously developing such items as office automation systems, user workstations, word processing systems, local networks, and all the rest. Many of the requirements originally stated for SAFE have become acknowledged requirements in the marketplace.

There is in addition a powerful collateral argument. An architecture that separates major functions and assigns them to dedicated machines can with modern technology couple machines from diverse vendors. For example, Rand operates a DEC 2060, a DEC VAX, two DEC PDP 11/70s, and a large IBM configuration. All the machines are linked by communication paths so, for example, text composed on a DEC system can be shipped electrically to the IBM for printing. The whole action is done directly from the user's console, and all intermachine file handling and protocols are handled by the systems. Thus one could imagine one machine dedicated to handling incoming electrical traffic and routing it to appropriate mailboxes. A second and distinct machine could handle all user terminal traffic and transactions; a third, the word processing and text composition features with automatic transfer of partial or complete files/records needed by the analyst for the document underway. Yet another machine could be dedicated to managing the overall database.....and on and on as system functions might dictate. There are a variety of advantages from such an approach.

- o Machines can be selected that are most appropriate for the function and/or happen to have commercially available software.
- o The software task is partitioned, and to the extent that new software must be written, the job will be easier.
- o As functions change and grow, or as new commercial products become available, only portions of the overall system need be changed, replaced, upgraded, etc. The same observation pertains as well to software. A similar observation clearly also pertains to growth in any one part of the system.
- o Standard interface arrangements exist for linking diverse machines together, so that only moderate amounts of software should be needed for overall system management.

The thrust of the argument so far is: take advantage of all the in-house personnel assets that you can lay hands on; choose an architecture that can exploit whatever commercially appropriate or specialized machines and/or software exist; look to the industrial world for innovative architectural possibilities; assay the TRW scene and attempt to exploit whatever designs/software/ personnel skills might be salvaged.

Consider now the SUL issue; it obviously needs a great deal of attention. The requirements of such an all-encompassing language will be extremely hard to design and implement; it may even be impossible as now envisaged. Based on our admittedly limited insights so far, it does not appear that individuals skilled in building systems that are friendly, smooth, forgiving, and otherwise attractive to users have been involved in language design. An example cited in the Selfridge report clearly shows that too many keystrokes will be required of users for common frequently done operations.

In this regard it is to be noted that a system architecture that divides functions among machines would permit components of a "SUL" to be commercially available designs; the word processing portion could even be CIA's own SCRIPT. A language for electronic message handling could be specially developed, a commercial design or even a developed elsewhere package. A similar observation applies to other parts of a user language; e.g., the part handling file transactions, the part for moving files or records among user functions, the part distributing incoming electrical messages, the part for building user profiles. In fact, it may be impossible to implement a comprehensive user language without segmenting it as just suggested. Finally, the design team must include a group of analysts to provide end-user viewpoints and preferences, knowledgeable individuals who are experienced in matching language to users, and systems people attuned to the interplay between language details and both hardware and software architecture.

System sizing might need re-examination. One feels that the number of computers proposed for the TRW design must reflect the aggregation of all the biggest parameters values ever imagined, i.e., maximum number of users combined with maximum interaction rates of all users all working on maximum size files. There must be internal communities of interest whose members are physically proximate and could be served by localized files and inter-terminal communication. One would be tempted to suggest a relook at the system sizing parameters, taking into account user-community neighborhoods and their localized needs. In part, the matter is also an architectural one.

There are some other issues on which we do not offer suggestions, but whose absence we note. First, there has been no mention of a test bed for exploring system design details. An undertaking as large as SAFE with so many untested concepts and approaches without an operational test vehicle is suicidal. The Block 2.5 system as a limited function operational system does not satisfy the purpose and does not allow any development cycle to take advantage of what will have been learned. In this same connection, there seems to be no mechanism to take advantage (as Selfridge has pointed out) of what has been learned in Pilot Mail and Interim SAFE.

Second, we encountered no discussion whatsoever of system security matters. Other than brief mention of encryption in connection with bus aspects, overall system security safeguards were not treated in the audit. We must ask: were they treated properly in the project or even at all?

Third, life cycle support of software will be a major aspect of SAFE's lifetime. Has it been considered in the project? Do system plans include an appropriate program development environment for the purpose?

The phrase "IBM compatible" bothers us. Does it mean hardware only? And if so, does it mean any plug compatible components? Or is it intended to mean IBM software as well? And if so, is it clear that IBM's present software is adequate to support the number of users and the variety of functions envisaged for SAFE? Does it imply a bet on a future IBM 3081-XA environment? From another point of view, would insistence on an "IBM environment," whatever it means, preclude taking advantage of (say) DEC machines and the extensive software packages that exist for them?

Here are a few other questions that seem to us important. The system, as described to us, seems to be understood as hardware arrangements. Ought it not be understood, described and characterized in terms of system software, information processes, and data flows?

A lot has been said about management shortfall on both sides, government and contractor. We'll add a new aspect: might not, or will not, development of C and D in parallel lead inevitably to divergence? To be sure, the four questions asked of the contractor will lead to some useful information, and it does try to do something useful with his money burn rate, but it must not be considered a definition of the future effort. At best, it can provide useful insights to salvageability of knowledge in the contractor. Alone though, it cannot be sufficient for a sound decision on the approach, system configuration, et al.

Here is a list of options which we offer as a menu at this time and without a recommendation.

- Start over. This might or might not also imply cancellation, depending on what is learned.

- If start over, then establish a small government SPO with experienced system acquisition people, use an experienced system support contractor to assist the SPO, look at existing capabilities that could fit into a different architecture (a UNIX environment for part of the system, perhaps coupled with the Programmer's Work Bench as terminal subsystems).

- Continue with TRW and Burroughs but restructure the project by establishing the small SPO and its support contractor, by reducing the TRW money expenditure rate to just that required for system design while trying to preserve the vested learning in the project, by insisting on good acquisition methodology, by simplifying requirements so that an incremental operational capability can be acquired soon, and finally, by establishing a baseline overall system design to work from.

- Cancel everything and wait until commercial developments can directly satisfy some or all of the SAFE requirements. Meanwhile expand Pilot Mail; install it at DIA; instrument both it and Interim SAFE to get user behavior and statistics to guide a future design.

- Do the best possible to give the analyst some prompt support by using existing commercial components that might stand alone with only rudimentary communication among them; e.g., use UNIX/PWB for terminal to various hosts together with SHELLS for special user functions. This approach could be extended by taking advantage of existing DBMS systems, intelligent terminals, local area networks, etc.

One last point which has been noted but which we reinforce.

> There must be an overall system architecture and design as the first priority of business. Until it exists, no implementation and coding for final operation can begin. Ideally, such a design would permit a project structure that will result in an early capability for the analyst with incremental enlargements, each of which build smoothly to what has already been achieved. Furthermore, the design ought to permit a phased approach in which lessons learned in earlier increments can be reflected in subsequent improvements.

Just to review the genesis of this letter: Ron and Kay provided notes to Willis who then synthesized a draft that was circulated to all four STABers for comment and modification. Since the discussion of April 15 was arranged after the meeting of April 13 concluded, Jim does not have the context behind the discussion above; he will submit his views separately. Thanks to word processing systems, telefax, Federal Express and even Express Mail, we made all this happen in a few days. If only everyone could have had a terminal on a common electronic message system.

Sincerely,

Willis H. Ware
Corporate Research Staff

WHW:dms